IN THE CLAIMS

1. (Previously Presented) A method of managing an arbitration queue having a plurality of queue entries comprising:

introducing entries into the queue at a first, highest order queue location;

determining if lower order queue locations are available;

if lower order queue locations are available, moving all higher order queue location contents to an adjacent lower order queue location per cycle until all lower order locations are filled;

associating each entry after placement in the queue to one of a plurality of groups, each of the plurality of groups having a different transaction parameter criteria;

determining which particular one of the plurality of groups to service based on the transaction parameter criteria;

servicing a particular entry in the particular one of the plurality of groups based on servicing criteria; and

moving all higher order queue entries, with respect to the particular entry being serviced, to an adjacent lower order location in the queue.

2. (Original) The method of claim 1, further comprising the step of marking a location of a serviced entry as idle.

3. (Original) The method of claim 2 wherein the moving step further comprises:

for higher order locations with respect to the idle location, writing the contents of higher order queue locations into adjacent lower order queue locations; and

for lower order locations with respect to the idle location, rewriting the current entry into the location.

4. (Original) The method of claim 1, further comprising the step of initializing all queue locations to an idle state prior to the step of introducing entries into the queue.

5. (Previously Presented) The method of Claim 1, wherein moving all higher order queue entries further comprises:

providing a plurality of registers corresponding to a number of entries in the queue, the plurality of registers being arranged in a linear array from a highest order register to a lowest order register;

for each register, selectively providing to each register an entry from that register or an entry from a higher order register.

6. (Previously Presented) The arbitration queue circuit according to claim 5, wherein entries are added to the queue via the highest order register.

7. (Original) The arbitration queue circuit according to claim 6, wherein the plurality of registers each have an entry output such that an entry can be removed from any location in the queue.

8. (Original) The arbitration queue circuit according to claim 7, wherein the plurality of registers includes 64 registers.

9. (Previously Presented) A computer system comprising:

a distributed shared memory system;

a plurality of processors generating transactions to said distributed shared memory system; and

a memory interface interposed between said distributed shared memory system and said plurality of processors, said memory interface having cache memory, a collapsible arbitration queue having a plurality of entry locations, and a memory arbitration processor for servicing transactions from said plurality of processors, the memory arbitration processor performing a memory arbitration scheme comprising:

placing transactions as entries in the arbitration queue;

associating entries after placement in the arbitration queue to one of a plurality of groups, each of the plurality of groups having a different transaction parameter criteria;

determining which particular one of the plurality of groups to service based on the transaction parameter criteria;

servicing a particular entry of the particular one of the plurality of groups;

marking a location of the particular entry in the arbitration queue as idle; and

collapsing the arbitration queue by bringing all higher order entries into adjacent lower order locations in the queue to fill the idle location.

10. (Original) The computer system according to claim 9, wherein the collapsing operation comprises:

for higher order queue locations with respect to the idle location, writing the contents of higher order queue locations into adjacent lower order queue locations; and

for lower order queue locations with respect to the idle location, rewriting the current entry into the location.

11. (Original) The computer system according to claim 9, wherein the plurality of entry locations includes a highest order location and a lowest order location, and wherein entries are added to the queue via the highest order location.

12. (Original) The computer system of claim 9, wherein the arbitration queue comprises:

a plurality of registers corresponding to the number of entries in the queue;

a plurality of 2:1 multiplexers interposed between said registers such that one multiplexer is interposed between a higher order register and a subsequent register, the output of said higher order register being coupled to a first input of said one multiplexer, the output of said subsequent register being coupled to a second input of said one multiplexer, an output of said one multiplexer being coupled to said subsequent register, and a mux control line being coupled to said one multiplexer to direct the contents of one of said first and second multiplexer inputs to the multiplexer output; whereby the mux control line associated with the higher order register and subsequent register determines whether the subsequent register is refreshed with its current contents or receives the contents of the higher order register.

13. (Original) The arbitration queue circuit according to claim 12, wherein the plurality of registers includes a highest order register and a lowest order register, and wherein entries are added to the queue via the highest order register.

14. (Original) The arbitration queue circuit according to claim 13, wherein the plurality of registers each have an entry output such that an entry can be removed from any entry in the queue.

15. (Original) The arbitration queue circuit according to claim 14, wherein the plurality of registers includes 64 registers.